



TITLE:

General Indexed Grammarと Monitored Pushdown Stack Acceptor (情報科学の数学的理論)

AUTHOR(S):

西沢, 輝泰

CITATION:

西沢, 輝泰. General Indexed Grammarと Monitored Pushdown Stack Acceptor (情報科学の数学的理論). 数理解析研究所講究録 1971, 123: 146-169

ISSUE DATE:

1971-09

URL:

<http://hdl.handle.net/2433/106499>

RIGHT:

General Indexed Grammar と Monitored Pushdown Stack Acceptor

京大 数研 西沢 輝 泰

§1. 序

general indexed grammar (西沢[2]) は、直観的に言えば、各変数が「記憶」をもつような context free grammar である。生成規則は、単に変数 X が string $u_0 Y_1 u_1 \dots Y_n u_n$ (Y_i は変数, u_i は terminal string) でおきかえられることを指定するだけでなく、 X の記憶内容に基づいて、各 Y_i の記憶内容も指定されなければならない。即ち、記憶内容の変換を表す関数 f_1, \dots, f_n を指定して、 X の記憶内容が α であるとき、各 Y_i の記憶内容は $f_i(\alpha)$ であるとするのである。変数 X もどのような string でおきかえるか、またおきかえる string 内の変数の記憶内容をよめる、記憶内容の変換の指定は、ともに、 X の記憶内容によって制御をうける。その制御は、記憶内容を有限種に類別しておいて、変数 X が記憶 α をもっているとき、

が類似にあれば, $X \rightarrow u_0 Y_1 u_1 \dots Y_n u_n$ なるおきかえが可能であり, そのとき, 記憶内容の変換は, 関数 f_1, \dots, f_n で指定される, という形でなされる。このとき,

$X \rightarrow u_0 Y_1 u_1 \dots Y_n u_n$ なるおきかえとそれのうける制御及び記憶内容の変換の指定は一体として,

$$(X, \gamma) \rightarrow u_0 (Y_1, f_1) u_1 \dots (Y_n, f_n) u_n$$

なる形式で表現される。これが, *general indexed grammar* の生成規則である。

例として, $\{a^n b^n c^n \mid n \text{ は正整数}\}$ を生成する *general indexed grammar* を与えてみる。変数として, S, A, B, C をとり, 各変数は任意の整数とその記憶内容としてとり得るものとする。記憶内容の類別, 即ちこの場合整数全体の類別としては, 0 である, と 0 でないという類別を考へ, 前者を γ で, 後者を γ' であるとする。変換 $f: \mathbb{Z} \rightarrow \mathbb{Z}$ を, $f(n) = n+1$ なる変換とし, $I: \mathbb{Z} \rightarrow \mathbb{Z}$ を恒等変換 $I(n) = n$ とする。生成規則として,

$$(1) \quad (S, \gamma) \rightarrow (S, f)$$

$$(2) \quad (S, \gamma') \rightarrow (S, f)$$

$$(3) \quad (S, \gamma') \rightarrow (A, I)(B, I)(C, I)$$

$$(4) \quad (A, \gamma') \rightarrow a(A, f^{-1})$$

$$(5) \quad (B, \gamma') \rightarrow b(B, f^{-1})$$

$$(6) \quad (C, 1) \rightarrow c(C, f^{-1})$$

$$(7) \quad (A, 3) \rightarrow a$$

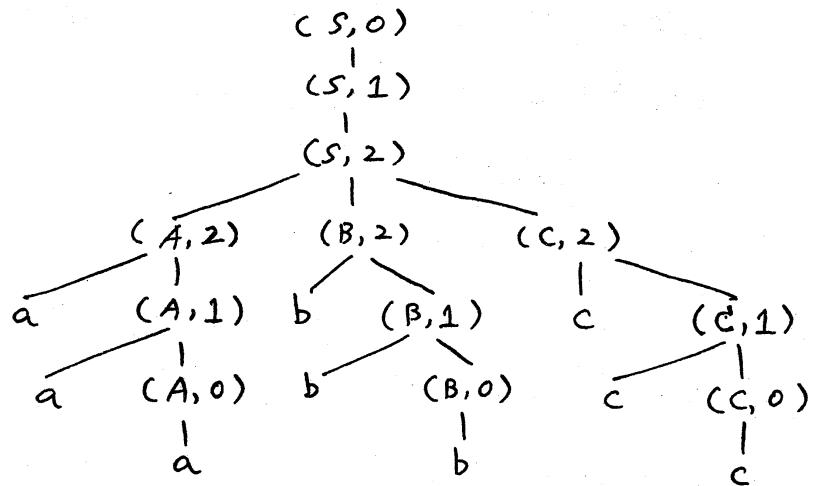
$$(8) \quad (B, 3) \rightarrow b$$

$$(9) \quad (C, 3) \rightarrow c$$

なる 9 個を与える。derivation の一例をとってみよう。
 sentence symbol は S , S の記憶内容は 0 から出発するもの
 としよう。記憶 α をもっている変数 X を (X, α) で表すこと
 にすると,

$$\begin{aligned} (S, 0) &\xRightarrow{(1)} (S, 1) \xRightarrow{(2)} (S, 2) \xRightarrow{(3)} (A, 2)(B, 2)(C, 2) \\ &\xRightarrow{(4)} a(A, 1)(B, 2)(C, 2) \xRightarrow{(5)} a(A, 1)b(B, 1)(C, 2) \\ &\xRightarrow{(6)} a(A, 1)b(B, 1)c(C, 1) \xRightarrow{(4)} aa(A, 0)b(B, 1)c(C, 1) \\ &\xRightarrow{(5)} aa(A, 0)bb(B, 0)c(C, 1) \xRightarrow{(6)} aa(A, 0)bb(B, 0)cc(C, 0) \\ &\xRightarrow{(7)} aaa(B, 0)c(C, 0) \xRightarrow{(8)} aaa bbb c(C, 0) \xRightarrow{(9)} aaabbbccc \end{aligned}$$

この生成過程を tree で表現すれば,



となる。

general indexed grammar は、言語の *syntax* の記述を、大ワケでは context free grammar で記述しておいて、それで記述しきれない部分を、変数のもっている記憶内容で check しようとするもので、現在プログラミング言語の記述に際してとられている手法にかんがみて、十分有用であると思われる。

ところで、記憶内容は、Turing machine においては *storage tapes* にかきこまれるわけであるが、general indexed grammar でもさしあたり各変数の記憶内容は *storage tapes* に記憶されるものと考えよう。*storage tape* の構造の代数的表現としては、T. Nishizawa [1] の, general counter structure を、少し手直しして用いる。定義と例は2節で述べるが、general counter structure は、pushdown stack の構造を一般化したものである。*storage tape* 一本は、2本の pushdown stack で簡単に simulate されるから、general counter structure は十分に一般的な構造であり、general indexed grammar において、各種の *storage tape* を用いたとものの共通した特性を一挙に解析することができよう。

general indexed grammar なる名称は、*storage tape* として pushdown stack を用いると、A.V. Aho [4] の indexed grammar と生成能力が同じになることと、各変数

の記憶内容は, その変数に添えられた index とも考えられることから名づけられた。

本稿では前半で 西沢 [2], [3] の結果を簡単に紹介し, 後半で, 任意の general counter structure τ に対し, 新たな general counter structure $\tilde{\tau}$ (τ -monitored pushdown stack) が得られて, τ -indexed grammar で生成される言語は, かつそれのみが, $\tilde{\tau}$ -counter machine (τ -monitored pda) で受理されることを示し, 任意の τ から $\tilde{\tau}$ が得られることを利用して, general counter structures の hierarchy と, これを反映した, indexed languages の hierarchy について論ずる。

§2. General Indexed Grammars

定義 1. general counter structure (以下 gcs と略記)

とは次のような系 $\tau = \langle \Gamma, \Pi, \Phi \rangle$ である。

- (i) Γ は空でない集合。
- (ii) Π は Γ の有限分割. ($\Pi = \{\gamma_1, \dots, \gamma_n\}$ と表すときは, γ_i は分割 Π の各ブロックであり, $\gamma_i \cap \gamma_j = \emptyset$ ($i \neq j$), $\bigcup_{i=1}^n \gamma_i = \Gamma$.)
- (iii) Φ は Γ 上の変換群で, 有限生成であって, かつ Γ 上推

移的。ただし、群演算は、 $(f, g) \mapsto g \circ f$, $g \circ f$ は $\alpha \in \Gamma$ に対し、 $(g \circ f)(\alpha) = g(f(\alpha))$ なる変換とする。

例. (1) $|\Gamma| = 1$ ($|\Xi| = 1$ でも同じ) なる gcs は *trivial gcs* と称し、 $|\Omega|$ で表す。

(2) $|\Gamma| < \infty$ ($|\Xi| < \infty$ でも同じ) なる gcs は、*finite gcs* と称する。

(3) (counter) $\begin{cases} \Gamma = \mathbb{Z} \text{ (整数すべての集合)} \\ \Pi = \{1, 0\}, \mathbb{Z} - \{0\} \\ \Xi = \{\bar{n} \mid n \in \mathbb{Z}\} \end{cases}$ ただし、 \bar{n} は $\bar{n}(m) = m + n$ なる \mathbb{Z} 上の変換。

この gcs を $|\frac{1}{2}|$ で表す。これは普通の counter である。

(4) (pushdown stack)

$\Gamma = \{a, b\}^D$ (ただし、 D は任意の集合として、 D^D で、 D で生成される自由群を表すものとする。)

$\Pi = \{ \langle e \rangle, \langle a \rangle, \langle b \rangle, \langle a^{-1} \rangle, \langle b^{-1} \rangle \}$, ただし、 e は Γ の単位元で、 $t \in \{e, a, b, a^{-1}, b^{-1}\}$ に対し、 $\langle t \rangle = \{x \in \Gamma \mid \text{end}(x) = t\}$; $\text{end}(x)$ は、 $\text{end}(e) = e$,

$\text{end}(t_1 \dots t_n) = t_n$, ただし $n \geq 2$ で各 t_i は a, b, a^{-1}, b^{-1} のいずれかで, 各 i について $t_{i+1} \neq t_i^{-1}$ とする, により定める.

$\Phi = \{ \bar{x} \mid x \in \Gamma \}$, ただし, \bar{x} は $\bar{x}(y) = yx$ なる Γ 上の変換 (right translation) を表す.

この $\text{geo} \langle \Gamma, \Pi, \Phi \rangle$ を *standard pushdown stack* と称す.

上記の例 (3), (4) とも, 分割 Π が, Γ の元唯1つの元からなるブロックをもつ. この性質は, *closure property* について考えるとき重要であって, $\text{geo } \tau = \langle \Gamma, \Pi, \Phi \rangle$ において, Π が, Γ の元唯1個からなるブロックを有するとき, τ は canonical type であると呼ぶことにする.

geo の合成として, 最も自然なものは, 2つの geo を並行して働かせる合成である.

2つの $\text{geo } \tau_i = \langle \Gamma_i, \Pi_i, \Phi_i \rangle$ ($i=1, 2$) に対し, $\text{geo } \tau = \langle \Gamma_1 \times \Gamma_2, \Pi_1 \times \Pi_2, \Phi_1 \times \Phi_2 \rangle$ を $\tau_1 \times \tau_2$ で表し, τ_1 と τ_2 の product geo と称す. ただしここで, $\Pi_1 \times \Pi_2$ は $\{ \xi_1 \times \xi_2 \mid \xi_1 \in \Pi_1, \xi_2 \in \Pi_2 \}$ なる $\Gamma_1 \times \Gamma_2$ の分割を表し, $(f_1, f_2) \in \Phi_1 \times \Phi_2$ は $(f_1, f_2)(\alpha_1, \alpha_2) = (f_1(\alpha_1), f_2(\alpha_2))$ なる $\Gamma_1 \times \Gamma_2$ 上の変換を表す.

定義2. $gcs \quad \tau = \langle T, \Pi, \Omega \rangle$ に対し, τ -indexed grammar とは, 次のような系 $G = \langle V, \Sigma, P, S, \alpha \rangle$ をいう。

(i) V は nonterminal alphabet (V の元を nonterminal とし, 変数とも呼ぶ), Σ は terminal alphabet, $S \in V$ は sentence symbol, α は Γ の特定の元で starting index と呼ばれる。

(ii) P は $(X, \xi) \rightarrow u_0 (Y_1, f_1) u_1 \cdots (Y_n, f_n) u_n$ (ただし, X, Y_i は変数, ξ は Π の元, 即ち分割 Π のブロック, u_i は terminal string, f_i は Ω の元) なる形式の rule からなる有限集合。($n=0$ のとき, この形式は $(X, \xi) \rightarrow u_0$ を表す。)

* τ -indexed grammar による言語の生成を定義する前に, 記号の使い方の約束をおく。

τ -indexed grammar $G = \langle V, \Sigma, P, S, \alpha_0 \rangle$ に言及するとき,

(1) X, Y, z 又はこれに添字のついたものは変数を表す。

(2) a, b, c " " " terminal " "。

(3) u, v, w " " " Σ^* の元 " "。
(terminal string)

(4) τ は特に構造が指定してなければ $\tau = \langle T, \Pi, \Omega \rangle$ として関連事項が記述される。

- (5) α, β, γ 又はこれに添字のついたものは Γ の元を表す。
 (6) ξ, η " Π の元 "。
 (7) f, g " 重の元 "。
 (8) φ, ψ " $(\Sigma \cup (V \times P))^*$ の元 "。

定義3. $G = \langle V, \Sigma, P, S, \alpha_0 \rangle \in \tau\text{-indexed grammar}$ とする。

(1) (X, α) と φ に対し, P の元 $(X, \xi) \rightarrow u_0 (Y_1, f_1) u_1 \cdots (Y_n, f_n) u_n$ で, $\alpha \in \xi$, $\varphi = u_0 (Y_1, f_1(\alpha)) u_1 \cdots (Y_n, f_n(\alpha)) u_n$ なる rule が存在するとき,
 $(X, \alpha) \vdash_G \varphi$ 又は単に $(X, \alpha) \vdash \varphi$ で表す。

(2) φ, ψ に対し, $\varphi = \varphi_1 (X, \alpha) \varphi_2$, $\psi = \varphi_1 \varphi_0 \varphi_2$ かつ $(X, \alpha) \vdash \varphi_0$ なる $\varphi_1, \varphi_2, (X, \alpha), \varphi_0$ が存在するとき,
 $\varphi \vdash_G \psi$ 又は単に $\varphi \vdash \psi$ で表す。

(3) $(\Sigma \cup (V \times P))^*$ の部分集合 A に対し,

$$D(A) = \{ \psi \mid \exists \varphi \in A, \varphi \vdash \psi \} \text{ とおく。}$$

各 φ に対し, $D^n(\varphi)$ ($n=0, 1, 2, \dots$) を,

$$\begin{cases} D^0(\varphi) = \{ \varphi \} \\ D^{n+1}(\varphi) = D(D^n(\varphi)) \end{cases}$$

で定め, $D^*(\varphi) = \bigcup_{n=0}^{\infty} D^n(\varphi)$ とおく。 $\psi \in D^*(\varphi)$

なるとき, φ は ψ を生成すると呼び, $\varphi \vdash_G^* \psi$ 又は単に

$\varphi \vdash^* \psi$ で表す。

(4) G で生成される言語 $L(G)$ とは, (S, α_0) から生成される terminal string 全体である。即ち,

$$L(G) = \{u \in \Sigma^* \mid (S, \alpha_0) \vdash^* u\}.$$

(5) τ -indexed grammar で生成される言語 L を, τ -indexed language と称し, τ -indexed languages すべてからなる言語類 L_τ で表す。

定義4. context free grammar の特殊化として, right linear (left linear, linear) grammar を得るのと全く同様にして, right linear (left linear, linear) τ -indexed grammar を定義する。right linear τ -indexed languages すべてからなる言語類 R_τ で表す。

次に [2], [3] で得られた結果 (ただし *印は [2], [3] による新結果) を列挙する。

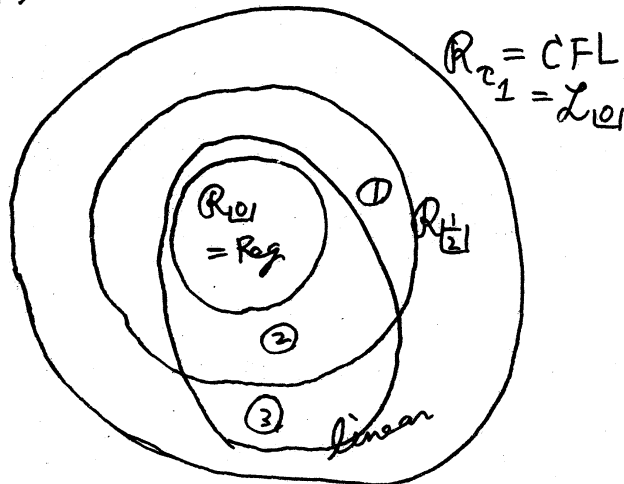
1. 以下で, τ_1 は standard pushdown stack, σ は任意の finite gcs とし,

$$\left. \begin{aligned} (1) \quad R_\sigma &= \{\text{regular set}\} \text{ (Reg)} \\ L_\sigma &= R_{\tau_1} = \{\text{context free language}\} \text{ (CFL)} \end{aligned} \right\} \text{自明}$$

$$(2) R_{\frac{1}{2} \times \frac{1}{2}} = L_{\frac{1}{2} \times \frac{1}{2}} = \{\text{recursively enumerable set}\} \text{ (RE)}$$

$$(3) L_{\tau_1} = \{\text{Ahoの indexed language}\} \text{ (Lindex)}$$

(4)



$$\textcircled{1} \{a^n c b^n a^m c b^m \mid n, m \in \mathbb{N}\}$$

$$\textcircled{2} \{a^n b^n \mid n \in \mathbb{N}\}$$

$$\textcircled{3} \{w c w^R \mid w \in \{a, b\}^*\}$$

(5)*

$$\textcircled{4} \{a^n b^n c^n \mid n \in \mathbb{N}\}$$

$$\textcircled{5} \{(w c)^3 w \mid w \in \{a, b\}^*\}$$

右のダイヤグラムは、
言語 ⑤ の位置 (証明略)

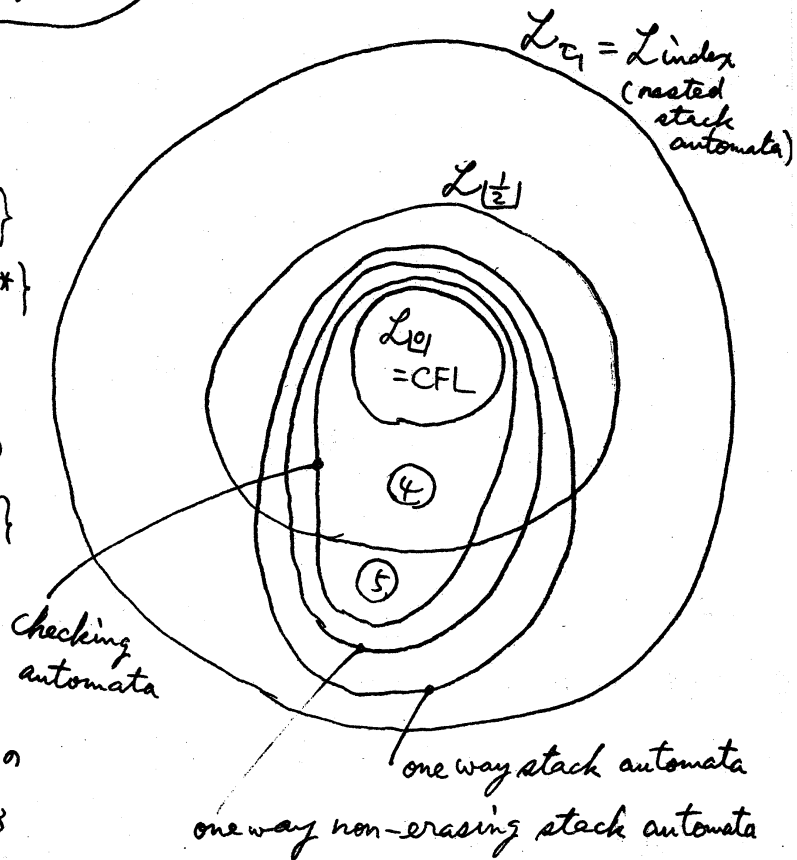
と $L_{\frac{1}{2}}$ の元 $\{a^n \mid n \in \mathbb{N}\}$

が checking automata

では受理されないこと、

及び S. Greibach [6] の

Theorem 4.1 (p. 213) から
得られる。



2. 以下で, τ は一般の gcs, τ_* は canonical gcs と
して,

- (1) $L_\tau \supset CFL$, $R_\tau \supset Reg$ (自明).
- (2) L_τ , R_{τ_*} は word reversal で closed.
- (3) L_τ (R_τ) は context free languages (regular sets) を代入する substitution に閉じて closed.
- (4) context free language (regular set) に, L_τ (R_{τ_*}) の元を代入したものは L_τ (R_{τ_*}) に属す.
- (5) L_{τ_*} は substitution で closed.
- (6) L_τ , R_{τ_*} は, union, concatenation, Kleene closure で closed. R_τ は union で closed.
- (7) L_τ , R_τ は NFT mapping (A.V. Aho [4]) で closed. 従って intersection with regular set で closed.
- (8) L_τ , R_{τ_*} は full AFL.
- (9) $L_\tau \not\supset RE$ ($R_\tau \not\supset RE \wedge R_\tau \supset \{\text{Dick language}\}$) なる, L_τ (R_τ) は intersection で閉じていない。特に, $L_{\lfloor \frac{1}{2} \rfloor}$, L_{index} はそうである。また $R_{\lfloor \frac{1}{2} \rfloor}$ は intersection で閉じていない。
- (10) L_τ の元は, 自然に拡張した standard function の minimal fixed point の一成分として特徴づけられる。

§ 3. gcs 間の simulation と, monitored pushdown stack

本節では \mathcal{L}_c の acceptor を与える準備をする。

Notation. (1) Γ^Γ (Γ は空でない集合) は, $(f, g) \mapsto g \circ f$ なる演算に関して monoid をなすが, この monoid を $M_\Gamma(\Gamma)$ で表す。

(2) gcs $\mathcal{C} = \langle \Gamma, \pi, \Xi \rangle$ に対し, $\mathcal{H}_{\mathcal{C}}$ で次のような $(\pi \times \Xi)^* \rightarrow M_\Gamma(\Gamma)$ の homomorphism を表す。

$$\mathcal{H}_{\mathcal{C}}((\xi, f))(\alpha) = \begin{cases} f(\alpha) & \text{if } \alpha \in \xi \\ \alpha & \text{if } \alpha \notin \xi \end{cases}$$

定義 5. $\mathcal{C} = \langle \Gamma, \pi, \Xi \rangle$, $\mathcal{C}' = \langle \Gamma', \pi', \Xi' \rangle$ を 2 つの gcs とする。

(1) \mathcal{C} の, \mathcal{C}' の representation とは, 次のような系 $\Theta = (\mu, \delta, \lambda)$ であり。

(i) μ は $\Gamma \rightarrow (\mathcal{P}(\Gamma') - \{\emptyset\})$ の mapping.

(ii) δ は $\pi \rightarrow (\mathcal{P}(\pi') - \{\emptyset\})$ の mapping で, $[\xi_1 \neq \xi_2] \supset [\delta(\xi_1) \cap \delta(\xi_2) = \emptyset]$ を満たす。

(iii) λ は $(\pi \times \Xi)^* \rightarrow (\pi' \times \Xi')^*$ の homomorphism.

(2) τ の τ' への representation $\textcircled{H} = (\mu, \delta, \lambda)$ に対し,
 $(\alpha, \alpha') \in \Gamma \times \Gamma'$ は, 次の条件を満たすとき, \textcircled{H} の下で *consistent* であるという。

$$(i) \alpha' \in \mu(\alpha), (ii) [\exists \beta \alpha \wedge \exists \beta' \alpha'] \supset [\exists' \in \delta(\exists)]$$

(3) τ の, τ' への representation $\textcircled{H} = (\mu, \delta, \lambda)$ は, 次の条件を満たすとき, τ の, τ' による *simulation* と呼ばれる。

(i) 任意の $\alpha \in \Gamma$ に対し, $\alpha' \in \Gamma'$ が存在して, (α, α') は \textcircled{H} の下で *consistent* .

(ii) 任意の $\alpha \in \Gamma, f \in \Pi, \exists \in \Pi$ に対して, (α, α') が *consistent* ならば, $(\mathcal{H}_\tau((\exists, f))(\alpha), \mathcal{H}_{\tau'}(\lambda((\exists, f)))(\alpha'))$ は *consistent* .

定義 6. *gcs* 間の関係 $<, \sim$ を次のように定める。

(1) $\tau < \tau'$ (τ は τ' で *simulate* される) \iff 適当な *finite gcs* σ により, τ の, $\sigma \times \tau'$ による *simulation* が存在する。

(2) $\tau \sim \tau'$ (τ は τ' に 同値) $\iff \tau < \tau' \wedge \tau' < \tau$

関係 $<$ について, 次のことが確かめられる。

(1) 任意の gcs τ と, 任意の $finite\ gcs$ σ について,
 $\sigma \prec \tau$.

(2) $\tau_i \prec \tau'_i$ ($i=1, 2$) ならば, $\tau_1 \times \tau_2 \prec \tau'_1 \times \tau'_2$.

(3) $\tau_1 \prec \tau_2$, $\tau_2 \prec \tau_3$ ならば, $\tau_1 \prec \tau_3$.

(4) $\tau_1 \prec \tau_2$ ならば, $L_{\tau_1} \subset L_{\tau_2}$, $R_{\tau_1} \subset R_{\tau_2}$.

(特に, $\tau_1 \sim \tau_2$ ならば, $L_{\tau_1} = L_{\tau_2}$, $R_{\tau_1} = R_{\tau_2}$)

さて, τ -monitored pushdown stack の定義にとりかか
 前に, 若干の Notation を定め, \prec , \sim について, もう少し詳
 しく調べる。

Notation. (1) 空でない集合 D に対する $D^\#$, 及び $x \in D^\#$ に対する $end(x)$ は §2 で述べた通り。

(2) 群重に対し, $e_{\text{重}}$ (又は単に e) は重の単位元を表す。

(3) 群重の部分集合 M に対し, M^{-1} で $\{m^{-1} \mid m \in M\}$ も, \overline{M} で $M \cup M^{-1} \cup \{e\}$ を表す。単なる空でない集合 D に対して, D^{-1} , \overline{D} 等の notation を用いた場合は, $D \in D^\#$ の部分集合とみなして用いたものと考えよ。

(4) D を空でない有限集合, $\tau = \langle \Gamma, \Pi, \text{重} \rangle$ を gcs , M を重の空でない有限部分集合, α を Γ の元として, 次のような gcs $\tilde{\tau} = \langle \tilde{\Gamma}, \tilde{\Pi}, \text{重} \rangle$ を $\otimes (D, \tau, M, \alpha)$ と表す。

$$(i) \quad \tilde{\Gamma} = (D \times \overline{M})^{\#}$$

(ii) Ξ は $\tilde{\Gamma}$ の right translations 全体のなす群. ($\tilde{\Gamma}$ と同一視する.)

(iii) $\varphi \in \tilde{\Gamma} \rightarrow \Xi$ の次のような homomorphism とする. (a) $\varphi((d, f)) = f$ for all $(d, f) \in D \times \overline{M}$, (b) $\varphi(t_1 \circ t_2) = \varphi(t_2) \circ \varphi(t_1)$ for all $t_1, t_2 \in \tilde{\Gamma}$.

$t \in (D \times \overline{M}) \cup (D \times \overline{M})^{-1}$ と $\xi \in \Pi$ に対し, $J(t, \xi)$ で $\{x \in \tilde{\Gamma} \mid \text{end}(x) = t, \varphi(x)(\alpha) \in \xi\}$ を表す. これにより, $\tilde{\Pi} = \{J(t, \xi) \mid \xi \in \Pi, t \in (D \times \overline{M}) \cup (D \times \overline{M})^{-1}\} \cup \{e_{\tilde{\Gamma}}\}$.

補助定理 7. $\tau = \langle \Gamma, \Pi, \Xi \rangle$, M, M' を Ξ を生成する Ξ の部分集合, α, α' を Γ の任意の元とする.

(1) D, D' を, 2個以上の元をもつ任意の有限集合とすれば, $\otimes(D, \tau, M, \alpha) \sim \otimes(D', \tau, M', \alpha')$

(2) τ が nontrivial gcs であれば, D, D' を任意の空でない有限集合として, $\otimes(D, \tau, M, \alpha) \sim \otimes(D', \tau, M', \alpha')$

(3) $\tau = \langle \{\alpha\}, \{\{\alpha\}\}, \{I\} \rangle$ は trivial gcs (101), D を 2個以上の元をもつ有限集合, D_0 を唯/1個の元からなる集合とすれば,

(a) $\otimes(D, \tau, \{I\}, \alpha) \sim \text{standard pushdown stack}$

$$(b) \quad \otimes(D_0, \tau, \{I\}, \alpha) = \underline{\frac{1}{2}}$$

定義 8. $gcs \tau = \langle \Gamma, \Pi, \Xi \rangle$ に対し, M は Ξ を生成する Ξ の有限部分集合, α は Γ の元, D は空でない有限集合とするとき, $gcs \otimes(D, \tau, M, \alpha)$ を τ -monitored pds (pushdown stack) と称す。

上の補助定理 7 によつて,

1. τ が nontrivial gcs なら, τ -monitored pds $\tilde{\tau}$ は, 同等なものをも同一視すれば 唯 1 つである。
2. $\underline{1}$ -monitored pds は 同等なものをも同一視すれば 唯 2 つある。1 つは $\underline{\frac{1}{2}}$ であり, 他の 1 つは standard pds に同等なものである。後者を $\underline{1}$ で表す。

補助定理 9. (1) $*\tau$ を nontrivial gcs, τ を一般の gcs として, $(*\tau \times \tau) \sim \tau \times *\tau$.

(2) $*\tau$ を nontrivial gcs として, $*\tau \sim \tau \times \underline{1}$

定理 10. (1) τ が nontrivial gcs なら, $\underline{1} < \tau$.

(2) τ_1, τ_2 が nontrivial gcs で, $\tau_1 < \tau_2$ なら, $\tilde{\tau}_1 < \tilde{\tau}_2$. 特に, $\tau_1 \sim \tau_2$ なら, $\tilde{\tau}_1 \sim \tilde{\tau}_2$.

(3) σ が nontrivial finite gcs ならば, $\sigma \sim \lfloor 1 \rfloor$.

定理 10(2) により, 次に定める $\lfloor n \rfloor$, $\lfloor n \frac{1}{2} \rfloor$ はそれぞれ
れ 同等なものを除いて 唯一つである。

定義 11.
$$\begin{cases} \lfloor 2 \rfloor = \lfloor 1 \rfloor^{\sim}, & \lfloor \frac{1}{2} \rfloor = \lfloor \frac{1}{2} \rfloor^{\sim} \\ \lfloor n+1 \rfloor = \lfloor n \rfloor^{\sim}, & \lfloor (n+1) \frac{1}{2} \rfloor = \lfloor n \frac{1}{2} \rfloor^{\sim}. \end{cases}$$

明らかに, $k \leq k'$ ならば $\lfloor k \rfloor < \lfloor k' \rfloor$ である。

§4. R_c, L_c の acceptor.

まず, R_c の acceptor は次のように自然に定義される。

定義 12. gcs $\sigma = \langle \Gamma, \Pi, \Sigma \rangle$ に対する c -counter machine
(c -CM) とは, 次のような系 $\mathbb{P} = \langle \Sigma, K, \delta, q_0, \alpha_0, F \rangle$
である。

(i) Σ は input alphabet, K は state alphabet, q_0 は
initial state ($q_0 \in K$), F は set of final states ($\subset K$),
 α_0 は starting counter content ($\alpha_0 \in \Gamma$).

(ii) δ は, $K \times \Pi \times (\{\epsilon\} \cup \Sigma)$ の各元に, $K \times \Pi$ の空でない有限部分集合を対応させる, $K \times \Pi \times (\{\epsilon\} \cup \Sigma) \rightarrow \mathcal{P}(K \times \Pi)$ の mapping. ただし, ϵ は monoid Σ^* の単位元を表す.

定義 13. $P = \langle \Sigma, K, \delta, q_0, \alpha_0, F \rangle$ を τ -CM とする.

(1) $K \times \Pi \times \Sigma^*$ 上の 2 項関係 \vdash_P を次のように定める.
 $(q, \alpha, u) \vdash_P (q', \alpha', u') \iff [\exists a \in \{\epsilon\} \cup \Sigma, \exists f \in \Pi; \delta(q, \xi, a) \ni (q', f) \text{ for } \xi \ni \alpha \wedge f(\alpha) = \alpha' \wedge u = a u']$

(2) \vdash_P の transitive, reflexive closure を \vdash_P^* とする.

(3) $T(P) = \{ u \in \Sigma^* \mid (q_0, \alpha_0, u) \vdash_P^* (q, \alpha, \epsilon) \text{ for some } (q, \alpha) \in F \times \Pi \}$ とし, P で受理 (accept) される set (または language) という. 何らかの τ -CM で受理される language を τ -CM language という.

定理 14. τ -CM language \iff right linear τ -indexed language.

定義 15. τ -monitored pushdown stack acceptor (τ -monitored pda) とは τ -CM である. τ -CM language

guage を τ -monitored pda language と呼ぶ。

定理 16. τ -indexed language \longleftrightarrow τ -monitored pda language (i.e. $L_\tau = R_\tau$).

(この定理は context free language \longleftrightarrow pda language の一般化である。)

定理 16 により, nested stack automaton (A.V. Aho [4], [5]) と \sqcup -monitored pda とが同等であることがわかる。これ自体, 興味ある結果である。

\sqcup -monitored pda は, 具体的には, 次のような 2本の pushdown stack (pds) D と S を storage tapes としてもつ, on-line nondeterministic Turing machine である。BPT, D に対しどのような symbol を pushdown 又は pop-up するかは, state と D, S の top symbols によって定まるが, S に対しどのような symbol を pushdown 又は pop-up するかは, そのとき D に対しどのような symbol が pushdown 又は pop-up されるかに完全に従属して定まる。しかもこの従属関係は, D に symbol A を pushdown することから S に symbol B を pushdown することであるならば, D から A を pop-up することから BPT S から B を pop-up することになる。

るような、判限の強い従属関係である。このような pda D と S との関係も、 S は D の slave である、と表現することにする。すると、 M-monitored pda といふ、具体的には、次のような $n+1$ 本の pda を storage tapes として持つ、on-line nondeterministic Turing machine である。即ち、 i 本目の pda が $i-1$ 本目の pda の slave, \dots , $i+1$ 本目の pda が i 本目の pda の slave, \dots , $n+1$ 本目の pda が n 本目の pda の slave である。又、 $\text{M}_{\frac{1}{2}}\text{-monitored pda}$ は、 M-monitored pda にさらにもう 1 本 storage tape として counter を補ったもので、この counter は $n+1$ 本目の pda の slave になつていふような Turing machine である。

§ 4. Hierarchy

前節定理 16 と § 2 の結果から、直ちに次の定理を得る。

定理 17.

$$\begin{array}{ccccccc}
 R_{\{0\}} & \subsetneq & R_{\frac{1}{2}} & \subsetneq & R_{\frac{1}{4}} & \subsetneq & R_{\frac{1}{8}} & \subsetneq & R_{\frac{1}{16}} & \subsetneq & R_{\frac{1}{32}} & \subsetneq & R_{\frac{1}{64}} \\
 \parallel & & & & \parallel & & \parallel & & \parallel & & \parallel & & \parallel \\
 \text{Reg} & & & & L_{\{0\}} & & L_{\frac{1}{2}} & & L_{\frac{1}{4}} & & L_{\frac{1}{8}} & & L_{\frac{1}{16}} \\
 & & & & \parallel & & \parallel & & \parallel & & \parallel & & \parallel \\
 & & & & \text{CFL} & & & & & & & & L_{\text{index}}
 \end{array}$$

$$R_{\lfloor 2\frac{1}{2} \rfloor} \subset \dots \subset R_{\lfloor n \rfloor} \subset R_{\lfloor n\frac{1}{2} \rfloor} \subset R_{\lfloor n+1 \rfloor} \subset \dots$$

$$\quad \quad \quad \parallel \quad \quad \parallel \quad \quad \parallel$$

$$\quad \quad \quad L_{\lfloor n-1 \rfloor} \quad L_{\lfloor (n-1)\frac{1}{2} \rfloor} \quad L_{\lfloor n \rfloor}$$

従って、

$$\lfloor 0 \rfloor \preceq \lfloor \frac{1}{2} \rfloor \preceq \lfloor 1 \rfloor \preceq \lfloor \frac{1}{2} \rfloor \preceq \lfloor 2 \rfloor \preceq \lfloor \frac{3}{2} \rfloor \prec \dots \prec \lfloor n \rfloor \prec \lfloor n\frac{1}{2} \rfloor \prec \lfloor n+1 \rfloor \prec \dots$$

Notation. alphabet Σ の各元 a に対し、 Σ の元でない新しい symbol $\#_a$ を、 $a \neq a'$ ならば $\#_a \neq \#_{a'}$ なるように対応させる。 $u, v \in \Sigma^*$ に対し $u^{(v)}$ を string を、
 $u^{(\varepsilon)} = \varepsilon$, $u^{(a)} = u \#_a u \#_a u \#_a u$ ($a \in \Sigma$), $u^{(xa)} = u^{(x)} \cdot u^{(a)}$ ($x \in \Sigma^*$, $a \in \Sigma$) により定める。

補助定理 18. $L_1 = \{ u \# (u c u^{(v)} \#)^2 u \mid v = a^{|u|} b^{|u|}, u \in \{a, b\}^* \}$ は $\lfloor 1 \rfloor$ -indexed language ではない。(ただし、 $\#$, c は $\#_a, \#_b$ と異なる 2 つの symbols とする。)

補助定理 19. $L_2 = \{ u \# (u c u c u c u^{(u)} \#)^2 u \mid u \in \{a, b\}^* \}$ は $\lfloor \frac{1}{2} \rfloor$ -indexed language ではない。(ただし、 $\#$, c については上記と同じ。)

補助定理 18, 19 の証明では、前者で $\{ u c a^{|u|} b^{|u|} \mid u \in \{a, b\}^* \}$

$\{a, b\}^*$ が context free language ではないことを, 後者では, $\{u \# u \# u \# u \mid u \in \{a, b\}^*\}$ が $\lfloor \frac{1}{2} \rfloor$ -indexed language ではないことを用いる。

$L_1 \in \mathcal{L}_{\lfloor \frac{1}{2} \rfloor}$, $L_2 \in \mathcal{L}_{\lfloor 2 \rfloor}$ は実際に grammar を構成することによって示すことができ, 次の定理が成立する。

定理 20. $\mathcal{L}_{\lfloor 2 \rfloor} \subsetneq \mathcal{L}_{\lfloor \frac{1}{2} \rfloor} \subsetneq \mathcal{L}_{\lfloor 1 \rfloor}$

系 21. $\mathcal{L}_{\lfloor 1 \rfloor} \subsetneq \mathcal{L}_{\lfloor \frac{1}{2} \rfloor} \subsetneq \mathcal{L}_{\lfloor 2 \rfloor} \subsetneq \mathcal{L}_{\lfloor 3 \rfloor}$ 。

定理 21 により, 我々は $\mathcal{L}_{\text{index}}$ より真に大きく, RE より真に小さい full AFL $\mathcal{L}_{\lfloor \frac{1}{2} \rfloor}$ を得ることができた。

次の問題が未解決である。

1. $\forall n \geq 2$ に対し, $\mathcal{L}_{\lfloor n \rfloor} \subsetneq \mathcal{L}_{\lfloor n \frac{1}{2} \rfloor} \subsetneq \mathcal{L}_{\lfloor n+1 \rfloor}$?
2. 1 が成立しない場合, $\exists n_0$, $\mathcal{L}_{\lfloor k \rfloor} \subset \mathcal{L}_{\lfloor n_0 \rfloor}$ for $\forall k$?
3. $\bigcup_{n=0}^{\infty} \mathcal{L}_{\lfloor n \rfloor} \subsetneq \text{RE}$?
4. $\mathcal{L}_{\lfloor n \rfloor}$ ($n \geq \frac{1}{2}$) に関する member-ship problem.
($\mathcal{L}_{\lfloor n \rfloor}$ に関しては, member-ship problem, emptiness problem, $\varepsilon \in L$? の 3 問題群は いずれか 1 つ が可解であれば 残りも可解である。)

参考文献

- [1] T. Nishizawa, Sequential Machines with General counters, Mem. Fac. Sci., Kyushu Univ., Ser A, vol. 24, no. 1, 94-99 (1970)
- [2] 西沢輝泰, General Indexed Grammars, 日本数学会応用数学科分科会予稿集 (1970, 4月)
- [3] ———, General Indexed Grammar に関する若干の補足, 日本数学会応用数学科分科会予稿集 (1970, 10月)
- [4] A. V. Aho, Indexed Grammars — An Extension of Context-Free Grammars, J. ACM, vol. 15, no. 4, 647-671 (1968)
- [5] ———, Nested Stack Automata, J. ACM, vol. 16, no. 3, 383-406 (1969)
- [6] S. Greibach, Checking Automata and One-Way Stack Languages, J. CSS, vol. 3, 196-217 (1969)